

KNN Variants for Text Segmentation considering Feature Similarity

Taeho Jo
President
Alpha AI Research
Cheongju, South Korea
tjo018@naver.com

Abstract—In this research, we propose the KNN variants which consider the feature similarities, as the approaches to the text segmentation. The initial KNN algorithm was previously modified so, and applied to the task, by mapping it to a binary classification. In the case of the numerical vector-based versions, we mention the three KNN variants: one where the selected nearest neighbors are discriminated by their similarities, one where the attributes are discriminated by their correlations with the categories, and one where the training examples are discriminated by their credits. The three KNN variants are modified into the versions which consider the feature similarities, as well as the initial KNN algorithm. This research is intended to improve further the text segmentation performance by modifying them so.

I. INTRODUCTION

The text segmentation is defined as the automatic segmentation of a text into subtexts based on their topic transitions. This task is usually applicable to a long text, and a boundary is set in the position where the topic is changed into another topic. The text segmentation is interpreted into the binary classification of each adjacent paragraph pair into boundary or non-boundary, and a machine learning algorithm is applied to the binary classification. The process of text segmentation is to partition a text into paragraphs, adjacent paragraph pairs by sliding the paragraph sequence with the two-sized window, each paragraph pair is classified so, and each boundary is put between paragraphs in the pair which is classified into boundary. A text is partitioned into subtexts based on the topic transition, and subtexts exist as independent texts.

This research is motivated by the successful case of applying the KNN algorithm which considers the feature similarities to the text segmentation. It was mapped into a binary classification of adjacent paragraph pairs, and the similarity metric between two numerical vectors which reflects both the feature similarities and the feature value similarities. The KNN algorithm was modified, using the similarity metric between numerical vectors, and applied to the text segmentation. The modified KNN algorithm was empirically validated as its better approach than the traditional version to the text segmentation. The KNN variants are modified into the versions which consider the feature similarities and applied to the text segmentation.

The idea of this research is to apply the KNN variants which are modified into the feature similarity-based versions to the text segmentation. In the previous work, the similarity metric between numerical vectors which considers the feature similarities and the feature value similarities, was defined, and adopted for modifying the KNN algorithm. We mention the three KNN variants which discriminate the nearest neighbors, the attributes, and the training examples, by their distances, their correlations with the target outputs, and their qualities, as the approaches to the text segmentation. The KNN variants are modified into the feature similarity-based versions and applied to the text segmentation which is a classification of two adjacent paragraphs into boundary or continuance. The text summarization is a binary classification of a single paragraph into summary or non-summary, whereas the text segmentation is a binary classification of two adjacent paragraphs into boundary or continuance.

Let us mention some benefits from this research. The poor discrimination among numerical vectors from their sparseness is avoided by proposing the similarity metric between numerical vectors, considering the feature similarity. The classification performance is upgraded by replacing the KNN algorithm by the KNN variants. Additionally, it is upgraded by modifying the KNN variants into the feature similarity-based version. This research is intended to provide the more recommendable approaches for implementing the text segmentation system.

Let us mention the organization of this paper. In Section II, we explore the previous works which are relevant to this study. In Section III, we describe in detail what is proposed in this research. In Section IV, we mention the significances of this research and the remaining tasks. This paper is composed of the four sections.

II. PREVIOUS WORKS

This section is concerned with the previous works which are relevant to this research. It is intended to expand the style of modifying the KNN algorithm to its three variants and apply them to the text segmentation. The directions of exploring previous works are derivation of KNN variants, modification, and application of the standard KNN algorithm to the text segmentation, and the cases of using the feature similarities for improving machine learning algorithms. The

distinguishable points of this research are derivation of three KNN variants from the standard version, modification of them with the feature similarities, and application of them to the text segmentation. This section is intended to explore previous works for providing the background for this research.

Let us explore the previous works on KNN variants. In 2001, the KNN variant where nearest neighbors are discriminated by their distances from or similarities as a novice item was applied to the text classification by Han et al. [1]. In 2008, MKNN (Modified K Nearest Neighbor) the KNN variants where training examples are discriminated by their validness, was proposed by Parvin et al. [3]. In 2018, the KNN variant where the attributes are discriminated by their correlations were proposed by Nababan and Sitompul [7]. However, this research uses different specific metrics for discriminating attributes, nearest neighbors, and training examples.

Let us explore the previous works on applying the modified version of KNN algorithm to the text segmentation. In 2017, it was initially proposed that the modified version of KNN algorithm should be applied to the text segmentation by Jo [4]. In 2019, the modified KNN algorithm was validated in the task by Jo [10]. In 2023, a journal article which describes in detail the modified KNN algorithm and its application to the text segmentation is prepared for its publication by Jo [11]. This research is based on the three previous works.

Let us mention the previous works on the cases of using the feature similarities as the basis for modifying the KNN variants in this research. In 2002, feature similarities were used for selecting some features for using unsupervised learning algorithms by Mitra et al. [2]. In 2018, feature similarities were used for modifying the AHC algorithm as an approach to text clustering by Jo [5]. In 2018, the KNN algorithm which was modified based on the feature similarities was used for classifying words [6]. Feature similarities are used for solving the sparseness in numerical vectors.

Let us mention the differences of this research from the previous works which are explored above. The KNN variants which are mentioned in the previous works do not consider the feature similarities, and in this research, we modify them into their versions which consider the feature similarities. As the expansion of [11], we modify and adopt the KNN variants for implementing the text segmentation system. The feature similarities are used for modifying the KNN variants as well as the KNN algorithm and the AHC algorithm as the approach to the task. The modified KNN variants will be described in detail in Section III.

III. PROPOSED WORKS

This section is concerned with what is proposed in this research. In Section III-A, we describe the process of encoding

a text into a numerical vector and the proposed similarity metric. In Section III-B, we describe the standard version of KNN algorithm where the proposed similarity metric is adopted. In Section III-C, we derive the three variants from the standard version. In Section III-D, we present the system architecture of the text segmentation system.

A. Encoding Process

This section is concerned with the process of encoding a text into a numerical vector. Words are used as features for encoding a text so, and some are selected from words in a corpus. The similarity between words is computed based on texts with their collocations and is used as the similarity between features. The similarities among features are considered for computing the semantic similarity between texts. In this section, we describe the process of encoding a text into a numerical vector and the process of computing the similarity between texts.

The process of encoding texts into numerical vectors is illustrated in Figure 1. The entire text collection is indexed into a list of words which are feature candidates. Some words are selected as the features by criteria among the feature candidates in the second step. The weights of the selected features in the text are computed as their relationships with the text, as elements of the numerical vector. Refer to [8] for studying the process and the selection criteria in detail.

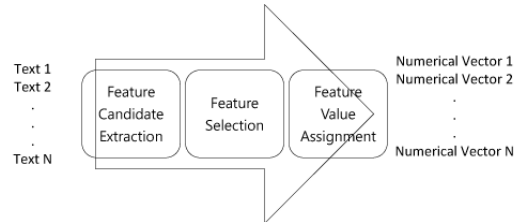


Figure 1. Process of Encoding Words into Numerical Vectors

The frame of computing the similarity between two numerical vectors is illustrated in Figure 2. The two d dimensional vectors and the d features are respectively notated respectively by $\mathbf{x} = [x_1 \ x_2 \ \dots \ x_d]$, $\mathbf{y} = [y_1 \ y_2 \ \dots \ y_d]$, and $f_1, f_2, \dots, f_d$. The feature similarity refers to the similarity between two features, which is notated by $\text{sim}(f_i, f_j)$. The feature value similarity is the similarity between two vectors, $\mathbf{x}$ and $\mathbf{y}$, such as cosine similarity. The similarity between words is computed by considering the two kinds of similarities.

Let us mention the process of computing the similarity between numerical vectors as the similarity between texts. The similarity between the features, f_i and f_j , is notated by $\text{sim}(f_i, f_j) = s_{ij}$. The similarity between the features is computed by equation (1),

$$s_{ij} = \text{sim}(t_i, t_j) = \frac{2 \times \#(t_i \wedge t_j)}{\#(t_i) + \#(t_j)}, \quad (1)$$

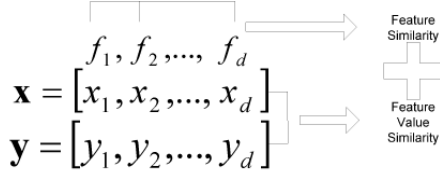


Figure 2. Frame of Computing Similarity between Numerical Vectors

where $\#(t_i \wedge t_j)$, is the number of texts which include both words, t_i and t_j , $\#(t_i)$ is the number of texts which include the word, t_i , and $\#(t_j)$ is the number of texts which include the word, t_j , and the similarity matrix to the d features is constructed as a $d \times d$ square matrix as follows:

$$S = \begin{pmatrix} s_{11} & s_{12} & \dots & s_{1d} \\ s_{21} & s_{22} & \dots & s_{2d} \\ \vdots & \vdots & \ddots & \vdots \\ s_{d1} & s_{d2} & \dots & s_{dd} \end{pmatrix}.$$

The similarity between the two numerical vectors, $\mathbf{x}$ and $\mathbf{y}$, is computed by equation (2),

$$\text{sim}(\mathbf{x}, \mathbf{y}) = \frac{\sum_{i=1}^d \sum_{j=1}^d s_{ij} x_i y_j}{d \|\mathbf{x}\| \|\mathbf{y}\|} \quad (2)$$

Equation (2) is used for computing the similarity between a novice text and a sample text in the KNN algorithm.

Let us make some remarks on the encoding process and the computation of the similarity between two texts. A text is encoded into a numerical vector by the feature candidate extraction, the feature extraction, and the feature value assignment. The similarity between features which are given as words is computed by equation (1). The similarity between numerical vectors which represent texts is computed by equation (2). In future, we consider computing the similarities among some features rather than all features for improving the computation speed.

B. Feature Similarity based KNN Algorithm

This section is concerned with the initial version of KNN algorithm which considers the feature similarities. The version was initially proposed as the approach to the text classification by Jo in 2019 [9]. The similarity metric between numerical vectors which was described in the previous section is used for computing the similarity between a novice vector and a training example. The labels of its nearest neighbors are voted with their identical weights for decoding the label of a novice vector. This section is intended to describe the initial version of the KNN algorithm from which its variants are derived.

The process of computing the similarity between a novice item and a training example is illustrated in Figure 3. The labeled words which are gathered as samples are encoded

into numerical vectors, and they are notated by a set $Tr = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\}$. A novice word is encoded into a numerical vector notated by $\mathbf{x}$. Its similarities with the numerical vectors which represent the sample words are computed by equation (2), as $\text{sim}(\mathbf{x}, \mathbf{x}_1), \text{sim}(\mathbf{x}, \mathbf{x}_2), \dots, \text{sim}(\mathbf{x}, \mathbf{x}_N)$. Some training examples which are most similar as the novice input are selected as its nearest neighbors.

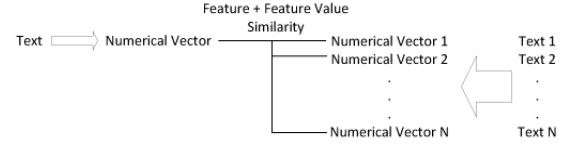


Figure 3. Similarities of Novice Input with Training Examples

In Figure 4, the process of selecting nearest neighbors by the ranking selection is illustrated. The training examples are sorted by the descending order of their similarities, after computing their similarities with a novice example. The training examples with their higher similarities with a novice example are selected as its nearest neighbors, as expressed in equation (3),

$$Nr = \text{Select}_k(Tr, \mathbf{x}), Nr \subseteq Tr \quad (3)$$

The set of nearest neighbors, Nr , is composed with elements as expressed in equation (4),

$$Nr = \{\mathbf{x}_1^{\text{near}}, \mathbf{x}_2^{\text{near}}, \dots, \mathbf{x}_k^{\text{near}}\} \quad (4)$$

The elements in the set, Nr , are used for voting their labels in the next step.

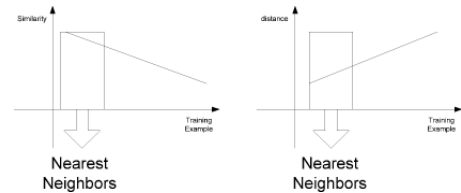


Figure 4. Selection of Most Similar or Least Distant Training Examples as Nearest Neighbors

The process of voting the labels of nearest neighbors for decoding the label of a novice word is illustrated in Figure 5. The predefined categories are notated by $c_1, c_2, \dots, c_m$, and the label of a nearest neighbor is notated by $c_j = \text{label}(\mathbf{x}_i^{\text{near}})$. The number of nearest neighbors which belong to the category, c_j , is notated by $\text{Count}(Nr, c_j)$. The label of the novice item, $\mathbf{x}$, is decided by equation (5),

$$\text{label}(\mathbf{x}) = \underset{j=1}{\text{argmax}}^m \text{Count}(Nr, c_j) \quad (5)$$

The process of deciding the label of a novice item by equation (5), is called voting.

Let us make some remarks on the initial version of KNN algorithm from which we derive some variants. It computes

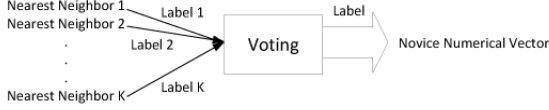


Figure 5. Voting of Labels of Nearest Neighbors for Deciding Label of Novice Input

the similarities of a novice item with the training examples. The training examples with their highest similarities with the novice item are selected as its nearest neighbors. The label of the novice item is decided by voting the labels of its nearest neighbors. The ranking selection is adopted for selecting the nearest neighbors from training examples, in this version.

C. KNN Variants

This section is concerned with the variants which are derived from the KNN algorithm which was covered in Section III-B. The difference from the traditional version is to adopt similarity metric which is expressed in equation (2) for computing the similarity between a novice item and a training example. In this section, we derive the three variants by discriminating nearest neighbors, attributes, or training examples. The three variants of KNN algorithm are adopted for implementing the text segmentation system. This section is intended to describe the three variants.

Let us mention the KNN variant which is derived by discriminating the nearest neighbors. A set of nearest neighbors is notated by $Nr = \{\mathbf{x}_1^{near}, \mathbf{x}_2^{near}, \dots, \mathbf{x}_k^{near}\}$, and its elements are assumed as $sim(\mathbf{x}, \mathbf{x}_1^{near}) \geq sim(\mathbf{x}, \mathbf{x}_2^{near}) \geq \dots \geq sim(\mathbf{x}, \mathbf{x}_k^{near})$. The weights, $w_1, w_2, \dots, w_k$, are assigned to the nearest neighbors, $\mathbf{x}_1^{near}, \mathbf{x}_2^{near}, \dots, \mathbf{x}_k^{near}$, following, $w_1 \geq w_2 \geq \dots \geq w_k$, and the total weights are computed for the category, c_j by equation (6),

$$CS(Nr, c_j) = \sum_{\mathbf{x}_i \in Nr} w_i \quad (6)$$

The label of the novice item, $\mathbf{x}$, is decided by equation (7),

$$label(\mathbf{x}) = \arg\max_{j=1}^m CS(Nr, c_j) \quad (7)$$

There are two ways of assigning weights to the nearest neighbors: exponentially decreasing way and arithmetic decreasing way as shown in Figure 6.

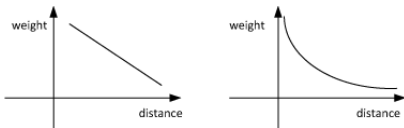


Figure 6. Discrimination over Nearest Neighbors

Let us mention the second KNN variant where the attributes are discriminated for computing the similarity

between a novice item and a training example as shown in Figure 7. If the number of training examples is N , and the dimension of numerical vector representing a training example is d , the attributes are notated as a set, $A = \{A_1, A_2, \dots, A_d\}$, and each attribute is viewed as a set of values, $A_i = \{x_{1i}, x_{2i}, \dots, x_{Ni}\}$. Each class is codified into its own integer, the correlation coefficient between an attribute A_i , and the classes is computed by equation (8),

$$r_i = \frac{\sum_{j=1}^N (x_{ji} - \bar{x}_j)(c_j - \bar{c})}{\sqrt{\sum_{j=1}^N (x_{ji} - \bar{x}_j)^2} \sqrt{\sum_{j=1}^N (c_j - \bar{c})^2}} \quad (8)$$

and the weight, which is assigned to the attribute, A_i , is set by the absolute value of the correlation coefficient, as expressed in equation (9),

$$w_i = |r_i| \quad (9)$$

The similarity between the novice input vector, $\mathbf{x}$ and $\mathbf{x}_i$, is computed by equation (10),

$$sim(\mathbf{x}, \mathbf{x}_i) = \frac{\sum_{j=1}^d \sum_{k=1}^d s_{jk} x_j w_k x_{ik}}{d \cdot \|\mathbf{x}\| \cdot \|\mathbf{x}_i\|}. \quad (10)$$

In this variant, equation (2) is replaced by equation (10) for computing the similarity.

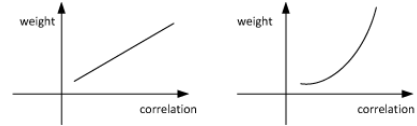


Figure 7. Discrimination over Attributes

Let us mention the third KNN variant where the training examples are discriminated for computing the similarity between a novice item and a training example and/or voting the nearest neighbors for deciding the label as shown in Figure 8. The hypersphere radius is used as the parameter, and for each training example, other training examples are taken within the radius from it as its nearest neighbors. The number of nearest neighbors and the number of training examples which are labeled identically to the training example, $\mathbf{x}_i$, are notated respectively by r_i and r_i^- , and the weight is computed by equation (11),

$$w_i = \frac{r_i^-}{r_i} \quad (11)$$

The similarity between the novice item, $\mathbf{x}$, and the training example, $\mathbf{x}_i$, is computed by equation (12),

$$sim(\mathbf{x}, \mathbf{x}_i) = \frac{w_i \sum_{j=1}^d \sum_{k=1}^d s_{jk} x_j x_{ik}}{d \cdot \|\mathbf{x}\| \cdot \|\mathbf{x}_i\|}. \quad (12)$$

and the total weight is computed for the category, c_j , by equation (13), in voting,

$$CS(Nr, c_j) = \sum_{\mathbf{x}_i \in Nr} w_i \quad (13)$$

Equation (12) and (13) are used respectively for computing the similarity between a novice item and a training example and voting the nearest neighbors for each category.

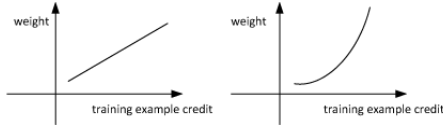


Figure 8. Discrimination over Training Examples

Let us make some remarks on the three variants of KNN algorithm which are proposed as the approaches to the text segmentation. In the first variant, the nearest neighbors are discriminated for voting their labels. In the second variant, the attributes are discriminated for computing the similarity between a novice input and a training example. In the third variant, the training examples are discriminated for voting the labels of the nearest neighbors and/or computing the similarity between numerical vectors. We may consider the trainable KNN algorithm which optimizes the weights of the nearest neighbors, the attributes, and the training examples for minimizing the training error.

D. System Architecture

This section is concerned with the architecture and the execution flow of the text segmentation system. In Section III-C, we describe the three KNN variants which consider the feature similarities and adopt them for implementing the text segmentation system. The initial KNN algorithm which is mentioned in Section III-B is replaced by its variants in the architecture of the text segmentation system which was previously proposed. The process of executing the system is to encode an adjacent paragraph pair into a numerical vector and classify it into boundary or continuance. This section is intended to describe the text segmentation system which is implemented in this study.

The process of gathering sample paragraph pairs and classifying a novice one, is illustrated in Figure 9. Because even a same paragraph pair may be classified differently depending on the domain, the task is called domain dependent classification. Sample paragraph pairs are gathered domain by domain, and each pair is labeled with boundary or continuance. The paragraph pairs are taken from texts which are tagged with their same domain, each paragraph pair is classified into boundary or continuance. Sample paragraph pairs which are labeled with one of the two categories are encoded into numerical vectors in each domain.

The entire architecture of the proposed text segmentation system is illustrated in Figure 10. A text is given is the input, and paragraph pairs are generated from the text by sliding the two sized window. The sample paragraph pairs in the boundary group and the continuance group and novice paragraph pairs which are generated from the input text are encoded into numerical vectors, in the encoding module.

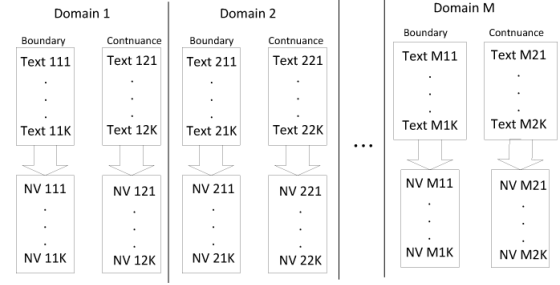


Figure 9. Preparation of Training Examples

The novice paragraph pairs are classified into one of the two categories in the similarity computation module and the voting module. The difference from the system architecture which was proposed in [10] is to replace the initial KNN algorithm by its variants.

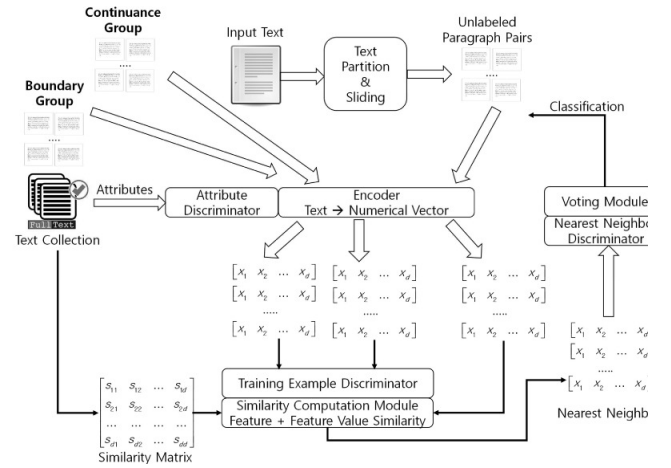


Figure 10. System Architecture

The execution process of the proposed system is illustrated as a block diagram in Figure 11. The sample paragraph pairs which are labeled with boundary or continuance are collected and encoded into numerical vectors. The paragraph pairs are generated from the input text by sliding the two-sized window on it, and they are encoded into numerical vectors. One among the three KNN variants is selected and applied to the classification of each paragraph pair into boundary or continuance. The boundary is marked between paragraphs in the pair which is classified into boundary.

Let us make some remarks on the proposed system whose architecture is illustrated in Figure 10. The M domains are predefined, in each domain, sample paragraph pairs each of which is labeled with boundary or continuance are prepared, and they are encoded into numerical vectors. The paragraph pairs which are generated from a text by sliding the two sized window are classified by one of the three KNN variants which are described in Section III-C. The difference from the

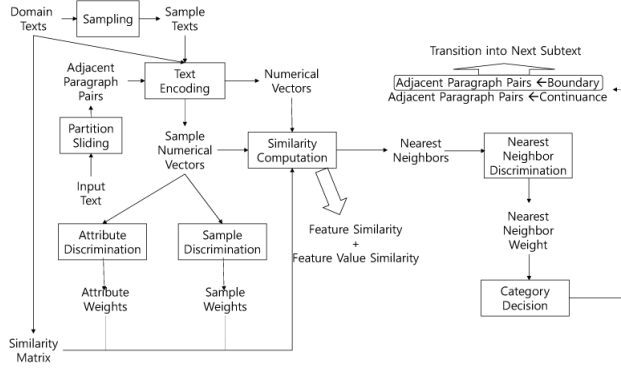


Figure 11. System Flow

previous version of text segmentation system is the ability to select one of the nearest neighbor discriminations, the attribute discriminations, and the training example discriminations. The better performance of text segmentation is expected by replacing the initial version by its variants.

IV. CONCLUSION

Let us mention the significances of this research as the conclusion. We apply the proposed similarity metric to the KNN variants as well as the standard version. We derive the three variants from the standard version by discriminating the nearest neighbors, the attributes, and the training examples. We design the text segmentation system by adopt the KNN variants with the proposed similarity metric. In the next research, we will implement it as a real system.

Let us mention the remaining tasks as the further discussions on this research. A text many be into a smaller dimensional vector by defining a word cluster as a feature, instead of an individual word. The deep learning algorithms may be modified by applying the proposed similarity metric between numerical vectors. We may apply the KNN variants which are modified with the proposed similarity metric to other tasks such as the text classification and the text summarization. The text segmentation may be implemented by adopting one among the proposed KNN variants as an independent program or a library module.

REFERENCES

- [1] E.H. Han, G. Karypis and V. Kumar, "Text Categorization Using Weight Adjusted k-Nearest Neighbour Classification" pp53-64, in *Advances in Knowledge Discovery and Data Mining. PAKDD 2001*, Berlin, Heidelberg:Springer, 2035, 2001.
- [2] P. Mitra, C. A. Murthy, and S. K. Pal. "Unsupervised feature selection using feature similarity", 301-312, *IEEE transactions on pattern analysis and machine intelligence* 24.3 2002.
- [3] H. Parvin, A. Hosein, and M.B. Behrouz, "MKNN: Modified k-nearest neighbor" *Proceedings of the World Congress on Engineering and Computer Science. Vol. 1. Newswood Limited*, 2008.
- [4] T. Jo, "K Nearest Neighbors for Text Segmentation with Feature Similarity", DOI: 10.1109/ICCCCEE.2017.7866706, *Proceedings of International Conference on Communication, Control, Computing and Electronics Engineering*, 2017.
- [5] T. Jo, "Clustering Texts using Feature Similarity based AHC Algorithm", 5993-6003, *Journal of Intelligent and Fuzzy Systems*, 35, 2018.
- [6] T. Jo, "Semantic Word Categorization using Feature Similarity based K Nearest Neighbor", 67-78, *Journal of Multimedia Information Systems*, 2018.
- [7] A.A. Nababan and O. S. Sitompul, "Attribute weighting based K-nearest neighbor using Gain Ratio", *Journal of Physics: Conference Series*, 1007, 1, 2018.
- [8] T. Jo, "Text Mining: Concepts and Big Data Challenge", Springer, 2019.
- [9] T. Jo, "Text Classification using Feature Similarity based K Nearest Neighbor", 13-21, *AS Medical Science*, 3, 2019.
- [10] T. Jo, "Content based Segmentation of News Articles using Feature Similarity based K Nearest Neighbor", 61-64, *The Proceedings of 19st International Conference on Information and Knowledge Engineering*, 2019.
- [11] T. Jo, "Content based Text Segmentation using Feature Similarity based K Nearest Neighbor", in Preparation, 2023.